Optimization of Input Parameters for the Flow-R Debris Flow Model

Documentation

by
Nele Rindsfüser
and
Basil Stocker

supervised by
Pascal Horton

Institute of Geography
University of Berne

2019

Contents

List of Figures

1 Introduction

1.1 Problem

Debris flows are complex phenomena with a wide range of material properties and can vary a priori in composition and flow behaviour from one event to another (Coussot & Meunier, 1996). The debris flow cannot be regarded as a Newtonian fluid due to its composition, which leads to some difficulties in the modeling and calculation of the events. Nevertheless, the mechanical and physical properties of debris flows have been well studied (Iverson, 1997) and have led to a large number of numerical models which can be used to calculate flow paths, heights and intensities of debris flows (Christen et al., 2012).

Due to the massively increased population and settlement areas in Switzerland during the 20th Century, the pressure on space is increasing and the potential damage caused by natural hazards is growing (*Naturgefahren: Entwicklung der Raumnutzung*, 2016). Due to population pressure in hazardous zones, the development of regional susceptibility maps for debris flows are important (Horton et al., 2013). To assess the regional susceptibility an empirical model called Flow-R has been developed.

1.2 The Flow-R Model

The model is called Flow-R for *Flow path assessment of gravitational hazards at a Regional scale. "It provides a substantial basis for a preliminary susceptibility assessment at a regional scale"* (Horton et al., 2013), using essentially a digital elevation model. To use the model two different steps based on the digital elevation model are required. First, the source areas are identified using morphological and user-defined criteria. Then the debris flows from the sources are propagated based on frictional laws and flow direction algorithms. The volume and mass of the debris flow is not considered (Horton et al., 2013). Flow-R is not suitable for modeling individual events. In order to evaluate the accuracy of the results and adjust the model parameters, it is best to compare the evaluated zone with specific events.

1.3 Goal

The canton of Berne has a natural hazards catalogue (Storme), which contains mapping of past natural hazard events. In order to test the model and find out the best parameters one can compare the model output with the existing mapping of a past event.

The goal of this project is to find the best values for the input parameters holmgren x and the reach angle for the Flow-R Model in specific catchments.

2 Method and Data

In order to find the optimal input parameters for Flow-R, an optimization program is produced. It should run the model with given parameters for a certain catchment, then compare the modeled event with actual data for past events in the same catchment. This comparison produces a quality measure. Or in other words: how well does the modeled event fit the extent of an actual event.

The optimization program then changes the input parameters and runs the model again and again, optimizing the quality measure. It is written in python using the *Python 2.7*-interpreter.

To run the presented optimization program, the Flow-R engine, a digital elevation model, a raster with a possible starting point of the debris flow, and the mapping of a past event from the natural hazard catalogue (ngkat) are needed. These files should ideally be structured as follows:

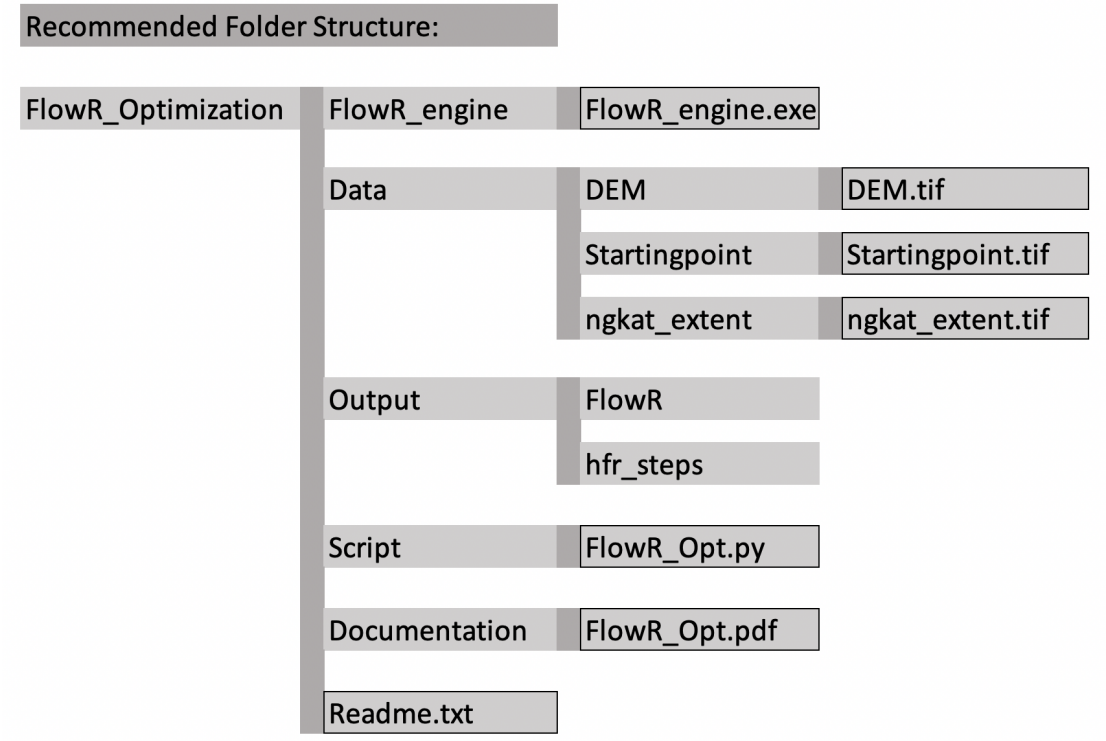

Figure 1: *Recommended folder-structure (screenshot).*

Further information on preprocessing these datasets can be found in the *readme*-file.

3 Code

To achieve the goal and realize the steps named above, the code is built as follows:

3.1 Import

In a first part, all the needed packages are imported and the ARCGIS-license is activated.

```
11  import subprocess
12  import arcpy
13  from scipy.optimize import brute
14
15  # Activate the ArcGIS Spatial Analyst extension license
16  arcpy.CheckOutExtension("Spatial")
17  arcpy.env.overwriteOutput = True
18
```

Figure 2: *Import of the packages needed for the optimization program and activation of ARCGIS-license (screenshot).*

3.2 Function Definition

Still in the opening of the code, the main function of this program is defined next. It is one big function, that does several different tasks and can be divided into multiple subparts. This function takes as argument a list of the two parameters holmgren x and reach angle that will be optimized. Its output is the quality measure, a hit-false-ratio, measuring the accuracy of the Flow-R-Model compared to the mapping of an actual event.

In the first subpart (Figure 3), all the parameters needed to run Flow-R are defined based on the function's argument and all the needed Flow-R-parameters are condensed into a single string. With the *subprocess.call* function, Flow-R is executed.

```
20  def FlowR_HFR(x):
21      """Runs FlowR and calculates a hit-false-ratio (hfr)"""
22      # run flow-r and calculate hit false ratio
23      holmgren_x = x[0]   # Parameters from function attribute
24      reach_angle = x[1]
25      run = str(x[0])+"_"+str(x[1])   # create name for folders based on changing input parameters
26      run_name = run_date+str(run)
27      # create string as input for Flow-R with all needed parameters
28      tot_string = [Flowr_path, "--dem=" + str(dempath), "--sources=" + str(startingpointspath), "--sources-val=ge1",
29                    "--spreading-algo=holmgren", "--holmgren-x=" + str(holmgren_x), "--holmgren-dh=" + str(holmgren_dh),
30                    "--friction-algo=reach-angle", "--reach-angle=" + str(reach_angle), "--max-velocity=" + str(max_velo),
31                    "--persistence-algo=weights", "--weights=default", "--output=1,3", "--output-path=" + str(outpath),
32                    "--threads-nb=8", "--log-level=2", "--run-name=" + str(run_name)]
33      # run Flow-R
34      subprocess.call(tot_string, shell=True)
35
```

Figure 3: *Run Flow-R (screenshot).*

Then (Figure 4), the resulting raster data is prearranged for the calculation of the hit-false-ratio. The raster from Flow-R is reclassified, so that cells within the range of the modeled debris flow contain the value 3, all others the value 0. Subtracting this layer from the actual event (from NGKAT, values 1 and 0) gives a new raster with the following values for every cell:

- $1 - 3 = -2$ true hit, actually affected cell also modeled as such
- $0 - 3 = -3$ false hit, model overestimates debris flow extent
- $1 - 0 = 1$ missed hit, model underestimates debris flow extent
- $0 - 0 = 0$ true negative, non-affected cell reproduced by model

```
32      # preparation for hit-false-ratio (hfr)
33      extent_flowr_pfad = outpath + "\\" + run_name + "\\" + run_name + "-extent-tot.tif" # path to the file created by Fl
34
35      extent_ngkat = arcpy.Raster(extent_ngkat_pfad)  # import of ngkat-raster
36      extent_flowr = arcpy.Raster(extent_flowr_pfad)  # import of Flow-R-raster
37      # no data to value 0
38      reclass_extent_ngkat = arcpy.sa.Con(arcpy.sa.IsNull(extent_ngkat), 0, extent_ngkat, "Value =1")
39      reclass_extent_ngkat.save(output + "\\" + run_name + "-reclass_extent_ngkat.tif")
40
41      # multiplie with multiplier to prepare calculation
42      ngkat_multiplier = 3
43      extent_ngkat_3 = reclass_extent_ngkat * ngkat_multiplier
44      extent_ngkat_3.save(output + "\\" + run_name + "-ngkat_multiplied.tif")
45
46      # subtract ngkat-raster from Flow-R-raster
47      dif_flowr_ngkat = extent_flowr - extent_ngkat_3
48      dif_flowr_ngkat.save(output + "\\" + run_name + "-dif_flowr_ngkat.tif")
49      dif_import_pfad = output + "\\" + run_name + "-dif_flowr_ngkat.tif"
50
```

Figure 4: *Prepare Hit-False-Ratio (screenshot).*

Last but not least, these values are counted over the whole area and the hit-false-ratio is calculated as follows (Figure 5):

$$1 - \left(\frac{\#\text{true hits}}{\#\text{true hits} + \#\text{false hits} + \#\text{missed hits}} \right)$$

This hit-false-ratio has values between 0 and 1 with lower values indicating a better quality. This is important, as the program later uses an minimization function.

The true negatives were left out of the calculation, as otherwise the extent of the chosen catchment would play a big role in calculating the hit-false-ratio.

```
    # search for hits and falses in dif_flowr_ngkat and count them
    my_dict = {row[0]: row[1] for row in arcpy.da.SearchCursor(dif_import_pfad, ['Value', 'Count'])}
    keys = my_dict.keys()
    if 0 in keys:
        count_0 = my_dict[0]
    else:
        count_0 = 0
    if 1 in keys:
        count_1 = my_dict[1]
    else:
        count_1 = 0
    if -2 in keys:
        count_2 = my_dict[-2]
    else:
        count_2 = 0
    if -3 in keys:
        count_3 = my_dict[-3]
    else:
        count_3 = 0
    # Hit false ratio:
    hfr = 1 - (count_2 / (count_1 + count_2 + count_3))
    return hfr                                          # returns hfr as result of the "FlowR_HFR(x)" function

```

Figure 5: *Calculate Hit-False-Ratio (screenshot).*

3.3 Working Directory and Data

Now, as the function is defined, the actual program begins. Therefore, the working directory and all the folders needed for the function above are defined. Additionally, the path to the raster data is defined.

```
# set Working Directory

work_dir = r"C:\Users\Peter_Muster\Documents\Fancy_Projects"


# Import Data and set Directories
Flowr_path = work_dir + r"\FlowR_Optimization\FlowR_engine\FlowR_engine.exe"
dempath = work_dir + r"\FlowR_Optimization\Data\DEM\DEM.tif"
startingpointspath = work_dir + r"\FlowR_Optimization\Data\Startingpoint\Startingpoint.tif"
extent_ngkat_pfad = work_dir + r"\FlowR_Optimization\Data\ngkat_extent\ngkat_extent.tif"
outpath = work_dir + r"\FlowR_Optimization\Output\FlowR"
output = work_dir + r"\FlowR_Optimization\Output\hfr_steps"

```

Figure 6: *Working directory and data paths for calculation (screenshot).*

3.4 Flow-R Parameters

The input parameters for Flow-R are defined in the following lines. First, the parameters that are not optimized (Holmgren dh and maximal velocity) are set, they stay fixed over the whole calculation. The run date is important, as folders and files are created. When running the program multiple times, the date should be changed, as a problem with already existing files will occur otherwise.

For the optimization function used, the ranges within which the function tries to optimize the input parameters, are also defined here.

```
# optimization

# Fixed parameters, don't change
holmgren_dh = 2
max_velo = 15
run = 0

# run_date format: "dd.mm.yyyy_", change to date of run
run_date = "31.01.2020_"

# ranges for the optimization function
rranges = (slice(1, 10, 1),slice(1, 20, 1))

```

Figure 7: *Definition of inputs into Flow-R (screenshot).*

3.5 Optimization

Finally, the optimization function is executed. It runs through the function defined above and tries to reach an optimal hit-false-ratio by adjusting the input parameters (Holmgren x and reach angle).

When finished, the program returns the optimal values of these parameters, as well as the hit-false-ratio reached.

```
# run the optimization function, which accesses the FlowR_HFR function again and again with changing parameters
solution = brute(FlowR_HFR, rranges, full_output=True)

# When the best hfr possible is found, give out the results to optimal holmgren x, reach angle, and the acquired hfr
print("The optimal input for holmgren x is:")
print(solution[0][0])
print("The optimal input for the reach angle is:")
print(solution[0][1])
print("The reached hfr:")
print(solution[1])

```

Figure 8: *Execution of optimization function (screenshot).*

4 Results

The optimization program was tested on the catchment of the Glyssibach in Brienz (BE), where a big debris flow occurred in 2005. The best reached Hit-false-ratio was around 0.644. This means that around 35% (as the hfr is inverse) of the mapped event area was hit by the model with the optimized two parameters Holmgren x and reach angle. The optimal input for Holmgren x is around 7.39 and for the reach angle around 7.02.

The hfr is still very high, as the mapping of the actual event was only executed in the lower, inhabited part of the catchment. This is clearly visible in Figure 9.

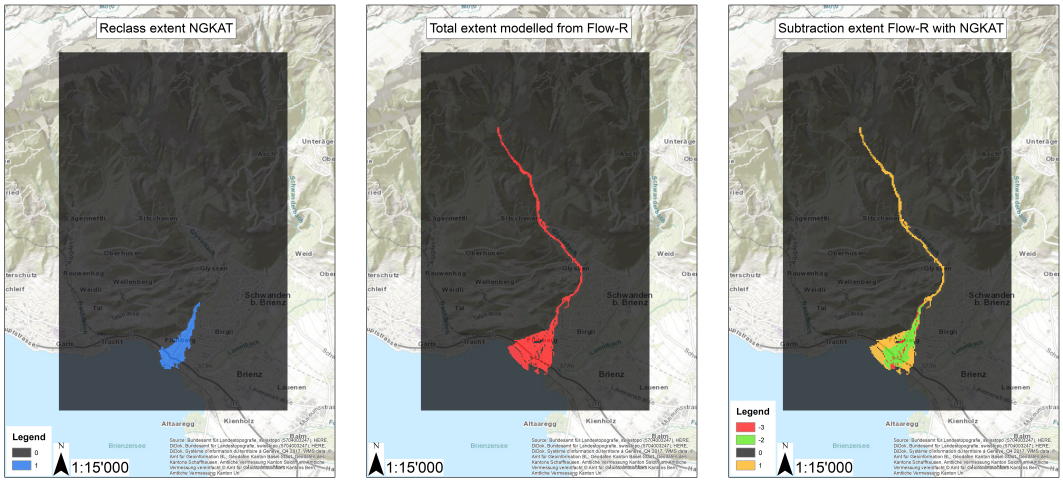

Figure 9: *Left: mapped extent of actual event; centre: modeled extent with optimized parameters; right: overlap, green indicating hits, red indication missed hits and orange indicating false hits (screenshots).*

5 Problems Encountered

One big problem was working with raster data, as some versions of Python and Arcpy could not handle them, while other versions would not work with Flow-R.

A different challenge was to create a system for the folder names. They had to be updated with every run in order to keep the program working. The solution was to include the optimizable variables into the name.

At first, test runs of the program did not go well due to the chosen DEM. The only obtainable one was a recent one from after the event and the subsequent hazard management measures. Thus, the model could not match the event's extent and the hfr was insufficient. Finally, a DEM from before the event could be obtained, and the results improved.

Finally, finding a fitting optimization tool was quite hard. The more elaborate tools seemed unable to screen the whole spectrum of possible values and spat out values very close to the starting points of the optimization. Therefore, the rather simple *Brute* approach from the *Scipy.optimize*-package was chosen.

References

Christen, M., Bühler, Y., Bartelt, P., Leine, R., Glover, J., Schweizer, A., ... Volkwein, A. (2012). Integral hazard management using a unified software environment: numerical simulation tool "RAMMS" for gravitational natural hazards. In (Vol. 1, p. 77-86). Grenoble - France.

Coussot, P., & Meunier, M. (1996). Recognition, classification and mechanical description of debris flows. *Earth-Science Reviews*, *40*, 209-227.

Horton, P., Jaboyedoff, M., Rudaz, B., & Zimmermann, M. (2013). low-r, a model for susceptibility mapping of debris flows and other gravitational hazards at a regional scale. *Natural Hazards and Earth System Sciences*, *13*, 869-885.

Iverson, R. M. (1997). The physics of debris flows. *Reviews of Geophysics*, *35*(3), 245.

Naturgefahren: Entwicklung der Raumnutzung. (2016). Retrieved 31.1.2020, from `https://www.bafu.admin.ch/bafu/de/home/themen/naturgefahren/fachinformationen/naturgefahrensituation-und-raumnutzung/naturgefahren--entwicklung-der-raumnutzung.html`